AD-A045 457 GENERAL ELECTRIC CO SYRACUSE N Y HEAVY MILITARY EQUI--ETC F/G 9/2
ANALYSIS OF A MULTI-SERVER, FINITE LENGTH QUEUE WITH FEEDBACK.(U)
SEP 77 G ARABADJIS

UNCLASSIFIED R77EMH9 NL

| OF |

AD
A045457

END
DATE
FILMED
11-77
DDC

'IS Distribution Center
SP 4-18, X7712
yracuse, New York 13221

AD A045457

(5)

GENERAL ELECTRIC

HEAVY MILITARY EQUIPMENT DEPARTMENT

(9) TECHNICAL INFORMATION SERIES,

Author	Subject Category	(14) No. R77EMH9
(10) G. Arabadjis	Queue Analysis (11)	[redacted] Sept 1977

Title

(6) ANALYSIS OF A MULTI-SERVER, FINITE LENGTH QUEUE WITH FEEDBACK.

Copies Available at	GE Class	1	No. of Pages
HMED TIS Distribution Center Box 4840 (CSP 4-18) Syracuse, New York 13221	Govt Class	Unclassified	~~32~~ (12) 34p.

Summary

This memo contains a performance analysis of a queueing model that arose in conjunction with a multi-microprocessor system with queue memories. The model consists of s-parallel processors fed by a common queue. Arriving tasks enter service immediately if a processor is available, or join a queue to wait for a processor, or become lost if the finite capacity queue is full. A completed task may leave the system or, with some probability, spawn another task which then goes to the end of the queue. The arrival stream is assumed Poisson and the service times are exponentially distributed.

This memo also contains curves of mean waiting time, fraction of lost tasks, fraction of time system is busy and mean busy period versus traffic intensity.

408969

Send to ___

GENERAL ELECTRIC COMPANY TECHNICAL INFORMATION

Within the limitations imposed by Government data export regulations and security classifications, the availability of General Electric Company technical information is regulated by the following classifications in order to safeguard proprietary information:

CLASS 1: GENERAL INFORMATION

Available to anyone on request.
Patent, legal and commercial review required before issue.

CLASS 2: GENERAL COMPANY INFORMATION

Available to any General Electric Company employee on request.
Available to any General Electric Subsidiary or Licensee subject to existing agreements.
Disclosure outside General Electric Company requires approval of originating component.

CLASS 3: LIMITED AVAILABILITY INFORMATION

Original Distribution to those individuals with specific need for information.
Subsequent Company availability requires originating component approval.
Disclosure outside General Electric Company requires approval of originating component.

CLASS 4: HIGHLY RESTRICTED DISTRIBUTION

Original distribution to those individuals personally responsible for the Company's interests in the subject.
Copies serially numbered, assigned and recorded by name.
Material content, and knowledge of existence, restricted to copy holder.

GOVERNMENT SECURITY CLASSIFICATIONS, when required, take precedence in the handling of the material. Wherever not specifically disallowed, the General Electric classifications should also be included in order to obtain proper handling routines.

GENERAL ELECTRIC COMPANY
HEAVY MILITARY EQUIPMENT DEPARTMENT
TECHNICAL INFORMATION SERIES

SECTION Information Processing Engineering

UNIT Information Systems Development Engineering

HMED ACCOUNTING REFERENCE 341

COLLABORATORS

APPROVED [signature] TITLE Mgr-Inf Systems Dev Eng LOCATION CSP 3, Room 16
Acting for R. E. Wengert

R77EMH9

MINIMUM DISTRIBUTION - Government Unclassified Material (and Title Pages) in G.E. Classes 1, 2, or 3 will be the following.

Copies	Title Page Only	To
0	1	Legal Section, HMED (Syracuse)
0	1	Manager, Technological Planning, HMED (Syracuse)
5	6	G-E Technical Data Center (Schenectady)

MINIMUM DISTRIBUTION - Government Classified Material, Secret or Confidential in G.E. Classes 1, 2, or 3 will be the following.

1	1	Classified Section, Electronics Park Library
1	0	Manager, Technological Planning, HMED (Syracuse)

ADDITIONAL DISTRIBUTION (Keep at minimum within intent of assigned G.E. Class.)

COPIES	NAME	LOCATION
5 (CLASS 1 ONLY)	DEFENSE DOCUMENTATI[illegible] CENTER	CAMERON STATION, ALEXANDRIA, VA. 22314
1	L. I. Chasen	P.O. Box 8555 Philadelphia, Pa., 19101
1	W. Adams	CSP4-48
1	T. Astemborski	FRP1-F3
1	D. Bahrs	CSP3-16
1	F. Binder	FRP1-F3
1	C. Blom	CSP5-M8
1	J. Breyer	FRP1-F2
1	J. Buchta	CSP4-58
1	E. Cabaniss	FRP1-K7
1	J. Chiasson	CSP3-15
1	F. DeBritz	FRP1-J7
1	O. Golubjatnikov	FRP1-J7
1	V. Grosso	CSP3-16
1	J.F. Jones	CSP5-K7
1	G. Kascha	CSP3-16
1	L. Knickerbocker	FRP1-N1
1	M. McCormack	CSP3-16
1	E. McCrohan	CSP3-16
2	D. Mott	CSP3-16

HMED 4024 REV 1/7/75

Copies	Name	Location
1	K. Olsen	CSP4-57
1	S. Parks	FRP1-57
1	P. Schuls	FRP1-F3
1	O. Shuart	CSP4-38B
1	R. Talham	FRP1-A5
1	F. Teillon	CSP5-K7
1	B. Viglietta	CSP4-57
1	R. Wengert	CSP3-16
50	G. Arabadjis (Class)	CSP3-16

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

SECTION I

INTRODUCTION

Reference 1 describes a multimicroprocessor architecture which consists of several parallel processors fed by a common queue. That paper also contains a performance analysis of the system's queueing behavior using an idealized model. This report contains an elaboration of that analysis for readers interested in the analytical details.

Preceding Page Blank - Filmed

SECTION II

THE MODEL

The queueing model is shown in Figure 2-1. It consists of s parallel processors (or servers) fed by a common queue which can accommodate up to q tasks (q is called the capacity of the queue). A task, when assigned to a server, is serviced to completion. Service time is assumed to be exponentially distributed with a mean of $1/\mu_o$ seconds, where μ_o is called the service rate. Upon completion of service, the task is released and the server instantaneously takes the next task in the queue to process. If the queue is empty, the server becomes idle. A completed task may leave the system or, with probability σ, spawn another task which then instantaneously goes to the end of the queue as shown by the feedback path in Figure 2-1.

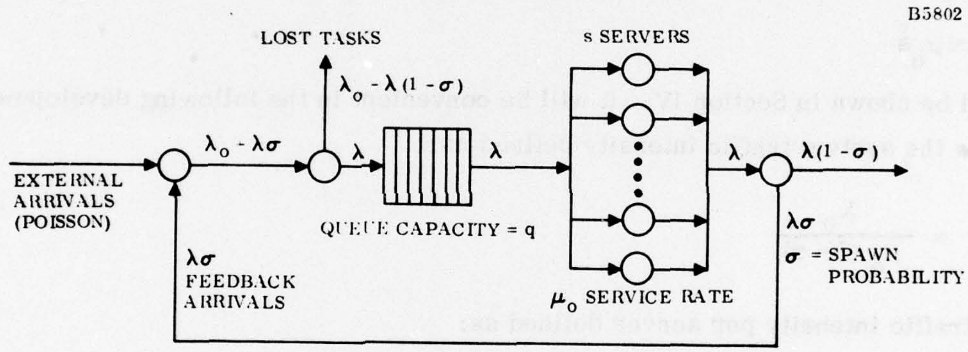

Figure 2-1. The System

If any task encounters a full queue, it becomes lost. The probability of a spawned task becoming lost is virtually zero since when the parent task completes service the server is released and a new task from the queue is instantaneously put into service. Thus, the spawning is synchronized with the decrease in queue size and the spawned task will almost always find at least one empty place in the queue. The probability that an external task arrives in the same time increment Δt as a spawned task is vanishingly small in this model.

The external arrivals to the system are assumed to be Poisson distributed with a mean arrival rate of λ_o arrivals/second. The average number of tasks processed per second by the servers is λ (the throughput), and $\lambda\sigma$ is the feedback arrival rate. Thus, the sum of the external and feedback arrivals, $\lambda_o + \lambda\sigma$, is the total task rate coming to the system. Since λ arrivals/second enter the queue, it follows that the rate at which tasks are lost is $\lambda_o + \lambda\sigma - \lambda$ tasks per second. The quantities λ_o, μ_o and σ are assumed to be known.

The queue mechanism described above implies that if one or more servers are idle then the queue must be empty. Conversely, if the queue is not empty then all servers must be busy. Since a full system consists of q tasks in the queue and s tasks in service (i.e., all servers busy), the capacity of the system is:

$$n = s + q \tag{2-1}$$

On the average, the s servers can complete at most $\mu_o s$ tasks per second. Therefore the average throughput λ cannot exceed this bound:

$$\lambda \leq \mu_o s \tag{2-2}$$

This will be shown in Section IV. It will be convenient in the following development to introduce the term traffic intensity defined as:

$$r_o = \frac{\lambda_o}{\mu_o (1 - \sigma)} \tag{2-3}$$

and the traffic intensity per server defined as:

$$\rho_o = \frac{r_o}{s} = \frac{\lambda_o}{\mu_o s (1 - \sigma)} \tag{2-4}$$

SECTION III

STATE PROBABILITIES

Queueing models with Poisson arrivals and exponential service time are about the simplest queues to analyze. Kleinrock's book (Ref. 2), Chapter 3, contains a readable exposition of different queueing models of this type. These are all referred to as M/M queues. The first "M" indicates that the arrival distribution is Poisson or, equivalently, that the interarrival time between two successive arrivals is exponentially distributed. The second "M" denotes that the service time is exponentially distributed. A queueing system with s parallel servers and queue capacity q is denoted as an M/M/s/q queue. In this report, reference will be made to s/q queues as shorthand for M/M/s/q.

Let N(t) denote the number of tasks in the system at time t. Similarly, let $N_q(t)$ and $N_s(t)$ denote respectively the number of tasks in queue and the number of tasks in service at time t. Then:

$$N(t) = N_s(t) + N_q(t) \tag{3-1}$$

$$N_s(t) = \begin{cases} N(t) & \text{for } 0 \le N(t) \le s \\ s & \text{for } s \le N(t) \le n \end{cases} \tag{3-2}$$

$$N_q(t) = \begin{cases} 0 & \text{for } 0 \le N(t) \le s \\ N(t) - s & \text{for } s \le N(t) \le n \end{cases} \tag{3-3}$$

Now define:

$$p_k(t) = P\{\, N(t) = k \,\}, \tag{3-4}$$

the probability that there are k tasks in the system at time t. There are three ways in which the system can be in state k at time t + Δ t:

1. The system was in state k-1 at time t and an external arrival occurred in time Δt with probability $\lambda_0 \Delta t$.

2. The system was in state k + 1 at time t and a task in service completed in time Δt, which did not spawn another task. This occurs with probability $N_s(t)\mu_o(1-\sigma)\Delta t$.

3. The system was in state k at time t and no task left or arrived to the system in time Δt, or a task completed service and spawned another task. This occurs with probability $1 - \lambda_o \Delta t - N_s(t)\mu_o(1-\sigma)\Delta t$.

The equations which describe these conditions for $0 \le k \le n$ are:

$$
p_k(t+\Delta t) = \begin{cases}
(1-\lambda_o \Delta t)\, p_o(t) + \mu_o(1-\sigma)\, p_1(t)\,\Delta t & k = 0 \\
\lambda_o p_{k-1}(t)\Delta t + [\,1-\lambda_o \Delta t - k\mu_o(1-\sigma)\Delta t\,]\, p_k(t) + (k+1)\mu_o(1-\sigma) p_{k+1}(t)\,\Delta t & 0<k<s \\
\lambda_o p_{k-1}(t)\Delta t + [\,1-\lambda_o \Delta t - s\mu_o(1-\sigma)\Delta t\,]\, p_k(t) + s\mu_o(1-\sigma) p_{k+1}(t)\,\Delta t & s \le k < n \\
\lambda_o p_{n-1}(t)\Delta t + [\,1-s\mu_o(1-\sigma)\Delta t\,]\, p_n(t) & k = n
\end{cases}
$$

Subtracting $p_k(t)$ from both sides of this equation, dividing by Δt, taking the limit $\Delta t \to 0$, and letting

$$
\dot{p}_k(t) = \lim_{\Delta t \to 0} \frac{p_k(t+\Delta t) - p_k(t)}{\Delta t}
$$

we obtain:

$$
\dot{p}_k(t) = \begin{cases}
-\lambda_o p_o(t) + \mu_o(1-\sigma) p_1(t) & k = 0 \\
\lambda_o p_{k-1}(t) - [\,\lambda_o + k\mu_o(1-\sigma)\,]\, p_k(t) + (k+1)\mu_o(1-\sigma) p_{k+1}(t) & 0<k<s \\
\lambda_o p_{k-1}(t) - [\,\lambda_o + s\mu_o(1-\sigma)\,]\, p_k(t) + s\mu_o(1-\sigma) p_{k+1}(t) & s \le k < n \\
\lambda_o p_{n-1}(t) - s\mu_o(1-\sigma) p_n(t) & k = n
\end{cases} \quad (3\text{-}5)
$$

This is a set of n + 1 simultaneous linear differential-difference equations whose solution is quite complex. The steady-state or equilibrium solution can be easily obtained, however, by setting $\dot{p}_k(t) = 0$ and solving the resultant difference equations:

$$
\begin{aligned}
& r_o p_o = p_1 \\
& (r_o + k)\, p_k = r_o p_{k-1} + (k+1)\, p_{k+1} && 0<k<s \\
& (r_o + s)\, p_k = r_o p_{k-1} + s\, p_{k+1} && s \le k < n \\
& s p_n = r_o p_{n-1}
\end{aligned} \quad (3\text{-}6)
$$

These n + 1 equations are not all independent however since any one can be derived from the remaining n equations. To obtain the unique solution we need the normalization condition

$$\sum_{k=0}^{n} p_k = 1 \tag{3-7}$$

which arises because the p_k's are probabilities of mutually exclusive events.

The solution to these equations is easily found by induction to be:

$$p_k = \begin{cases} \dfrac{r_o^k}{k!} \, p_o & 0 \le k \le s \\[2ex] \dfrac{r_o^k}{s^{k-s}\, s!} \, p_o & s < k \le n \end{cases} \tag{3-8}$$

where:

$$p_o = \left\{ \sum_{k=0}^{s} \frac{r_o^k}{k!} + \frac{r_o^{s+1}}{s!\,(s - r_o)} \left[1 - \left(\frac{r_o}{s} \right)^q \right] \right\}^{-1} \tag{3-9}$$

p_o is the probability that the system is empty; p_n is the probability that the system is full.

Preceding Page Blank - Not Filmed

SECTION IV

MEAN NUMBER OF TASKS

From the state probabilities several interesting quantities can be found. The mean number of tasks in the system is:

$$\overline{N} = \sum_{k=0}^{n} k\, p_k = \overline{N}_q + \overline{N}_s \tag{4-1}$$

where $\overline{N}_q$ and $\overline{N}_s$ are the mean number of tasks in queue, and the mean number in service, respectively. These are given by:

$$\overline{N}_q = \sum_{k=s+1}^{n} (k-s)\, p_k \tag{4-2}$$

$$\overline{N}_s = \sum_{k=0}^{s} k\, p_k + s \sum_{k=s+1}^{n} p_k \tag{4-3}$$

Preceding Page Blank - Not Filmed

SECTION V

THROUGHPUT, WAITING TIME AND RESPONSE TIME

The throughput, λ is the mean number of tasks serviced per second, which is simply:

$$\lambda = \mu_o \overline{N}_s \tag{5-1}$$

Since the mean number in service, $\overline{N}_s$, cannot exceed the number of servers, the throughput inequality given by Equation 2-2 is proved.

The mean response time T is the average time it takes a task to go through the system. T is composed of the wait time in queue, W, and the service time $1/\mu_o$;

$$T = W + \frac{1}{\mu_o} \tag{5-2}$$

T can be found from Little's theorem (Ref. 2). On the average, an arriving task should find the same number of tasks ($\overline{N}$) in the system as it leaves behind upon departure. The latter is merely the throughput λ times T, so that:

$$T = \frac{\overline{N}}{\lambda} = \frac{\overline{N}}{\mu_o \overline{N}_s} \tag{5-3}$$

Figure 5-1 shows plots of the normalized waiting time

$$\mu_o W = \frac{\overline{N}_q}{\overline{N}_s} \tag{5-4}$$

for various values of q and s versus ρ_o. The designator with each curve indicates the value of s and q, e.g., 4/8 indicates a four-server system with a queue capacity of eight. The normalized response time, $\mu_o T$, can be obtained from these curves by adding one (see Equation 5-2)). Note that increasing q and s such that their ratio q/s remains constant has the effect of reducing the response time for small ρ_o and increasing the response time for ρ_o moderately larger than unity. This increase occurs because relatively fewer tasks get lost as the queue capacity is proportionately increased.

For small ρ_o the curves approach straight lines on log-log paper. Furthermore, for given s, all curves approach the same straight line independent of q. Intuitively this is not surprising since at low traffic intensities the queue is almost never full when a task arrives.

B5803

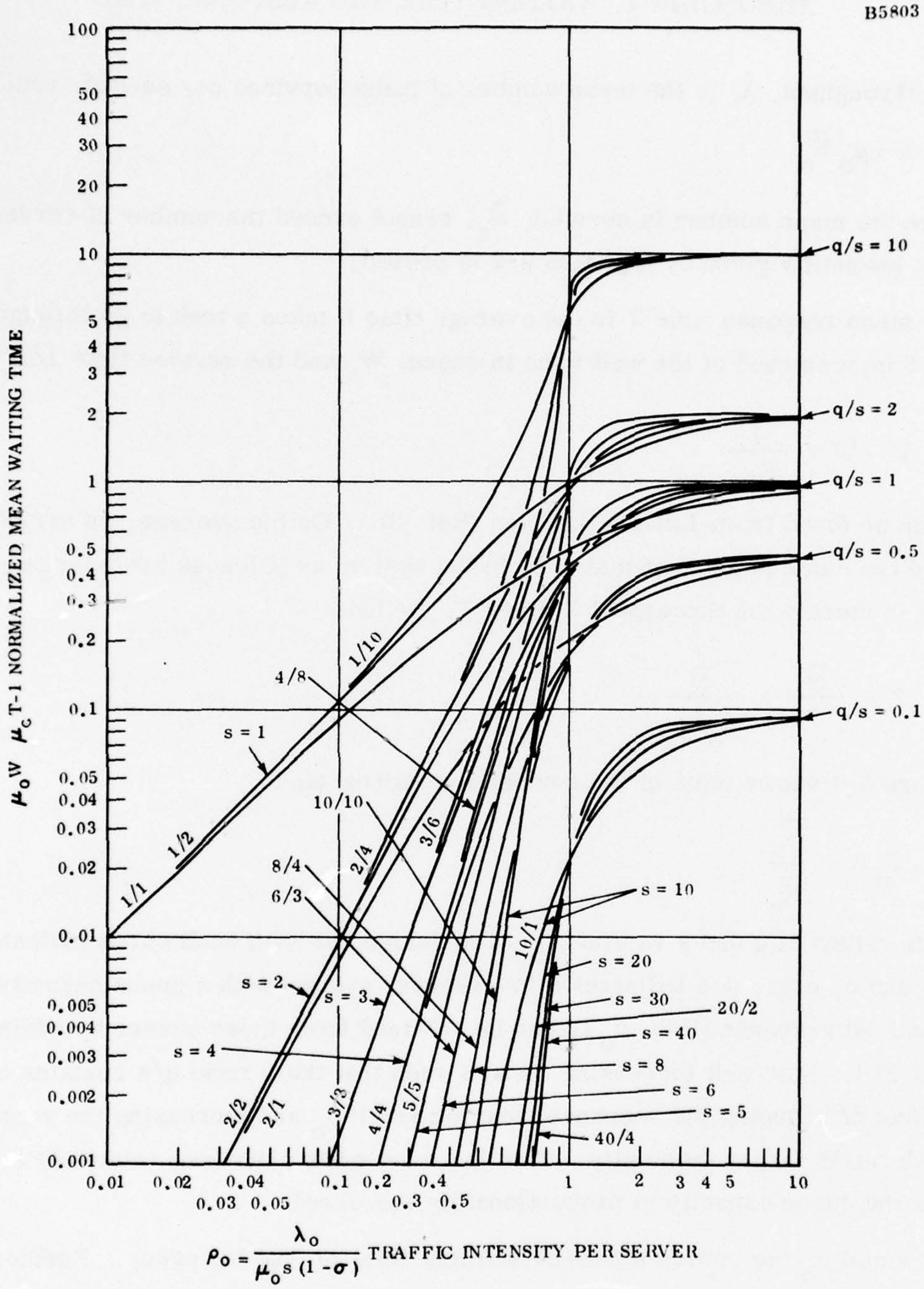

Figure 5-1. Mean Waiting Time vs Traffic Intensity

It is not difficult to derive an approximate expression for the waiting time valid for low traffic intensities. From equations (4-2), (4-3) and (5-4) we have

$$\mu_o W = \frac{p_{s+1} + 2\,p_{s+2} + \dots + q\,p_n}{p_1 + 2\,p_2 + \dots + s\,p_s + s\,(p_{s+1} + \dots + p_n)}$$

For small ρ_o, p_k decreases rapidly with increasing k. Therefore taking only the first terms in the numerator and denominator above yields (using Equation (3-8))

$$\mu_o W \approx \frac{p_{s+1}}{p_1} = \frac{r_o^{\,s+1}\, p_o}{s \cdot s!}\;\frac{1}{r_o p_o} = \frac{r_o^{\,s}}{s \cdot s!}$$

$$= \frac{s^{s-1}}{s!}\,\rho_o^{\,s} \qquad \text{For small } \rho_o \qquad (5\text{-}5)$$

For large ρ_o the curves approach a plateau whose value depends only on the ratio q/s. In fact

$$\mu_o W \approx \frac{q}{s} \qquad \text{For large } \rho_o \qquad (5\text{-}6)$$

Intuitively this result is reasonable. Under high traffic conditions an arriving task that joins the queue will find q-1 tasks in queue and s tasks in service. On the average it will take the s parallel servers $q/s\mu_o$ seconds to perform this work, hence Equation (5-6) follows.

Preceding Page Blank - Not Filmed

SECTION VI

LOST TASKS

From Figure 2-1, the total task arrival rate to the queue (external and feedback arrivals is $\lambda_o + \lambda\sigma$. The rate at which the tasks enter the system (i.e., become admitted to the queue) is λ, the throughput. Therefore, the rate at which tasks become lost is the difference between the input and throughput arrival rates, $\lambda_o + \lambda\sigma - \lambda$. Let L be the fraction of lost tasks:

$$L = \frac{\lambda_o - \lambda(1-\sigma)}{\lambda_o + \lambda\sigma} \tag{6-1}$$

Using Equations (2-3) and (5-1) this becomes

$$L = \frac{(1-\sigma)(r_o - \overline{N}_s)}{(1-\sigma)\, r_o + \sigma \overline{N}_s} \tag{6-2}$$

Figures 6-1 and 6-2 show plots of the percentage of lost tasks versus the per processor traffic intensity ρ_o. Figure 6-1 is for the case q = s and Figure 6-2 is for the case q = 2s for various values of s and σ. Figures 6-3 and 6-4 show plots of L versus the queue capacity q for various values of s at $\sigma = 0$ and $\sigma = 0.9$.

A response time $\mu_o T \approx 1$ for values of ρ_o up to 0.6 implies that a server is nearly always available to service the next incoming task. At traffic intensities above $\rho_o = 0.6$, the effects of queueing tasks become important very rapidly and the percentage of lost tasks (L) becomes high. Figure 6-3 shows that L is very sensitive to queue capacity. For example, increasing q from 5 to 8 (at $\rho_o = 0.5$) reduces the loss rate by an order of magnitude.

It is also interesting to note the behavior of the lost task percentage with spawn probability σ. Keeping other parameters constant, increasing the spawn probability substantially decreases the percentage loss. The reason for this decrease of task loss with increasing σ is primarily due to the assumed characteristics of the queueing model. As discussed earlier, a spawned task is synchronized with a decrease in queue size and thus is seldom lost.

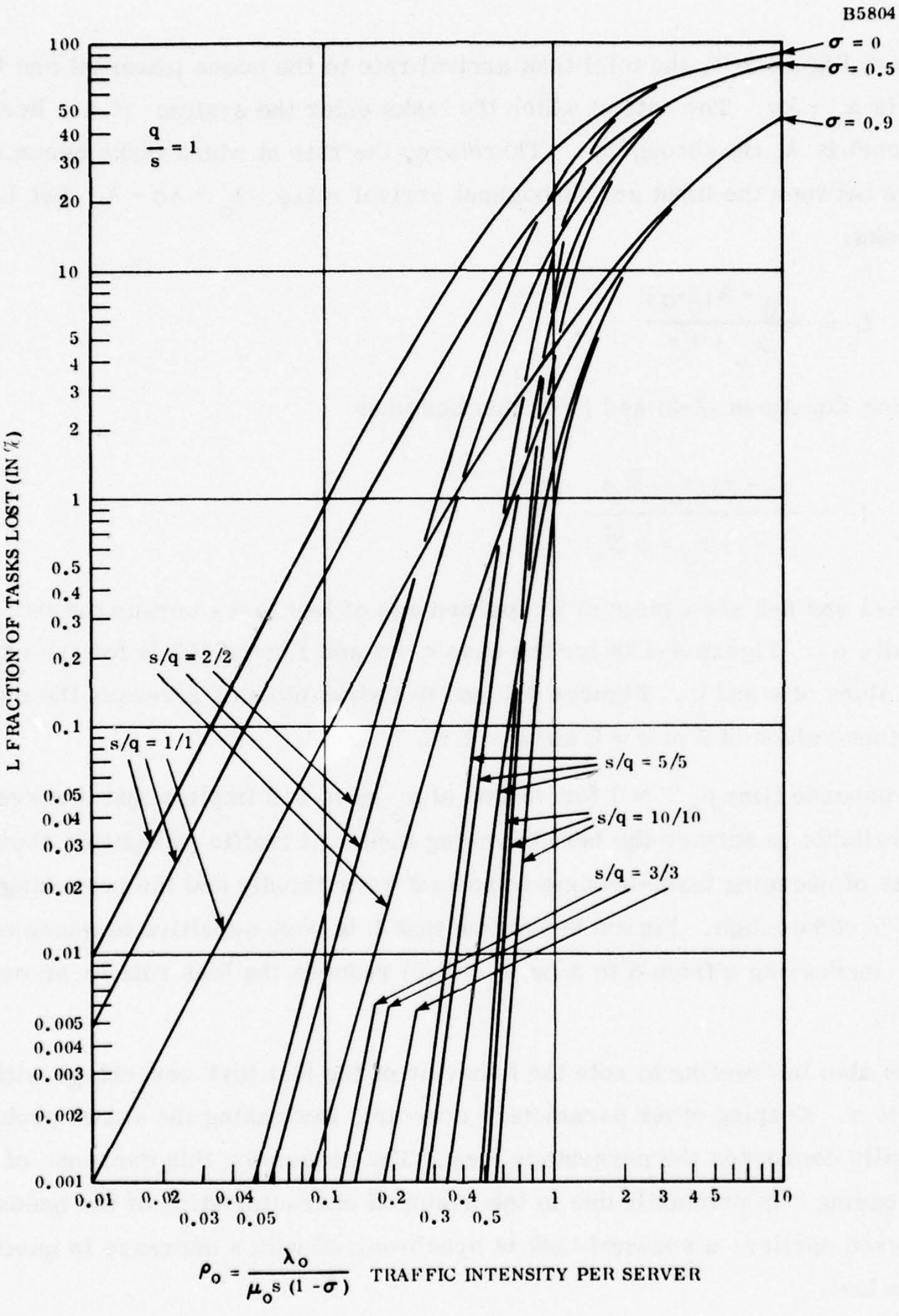

Figure 6-1. Fraction of Lost Tasks; q/s = 1

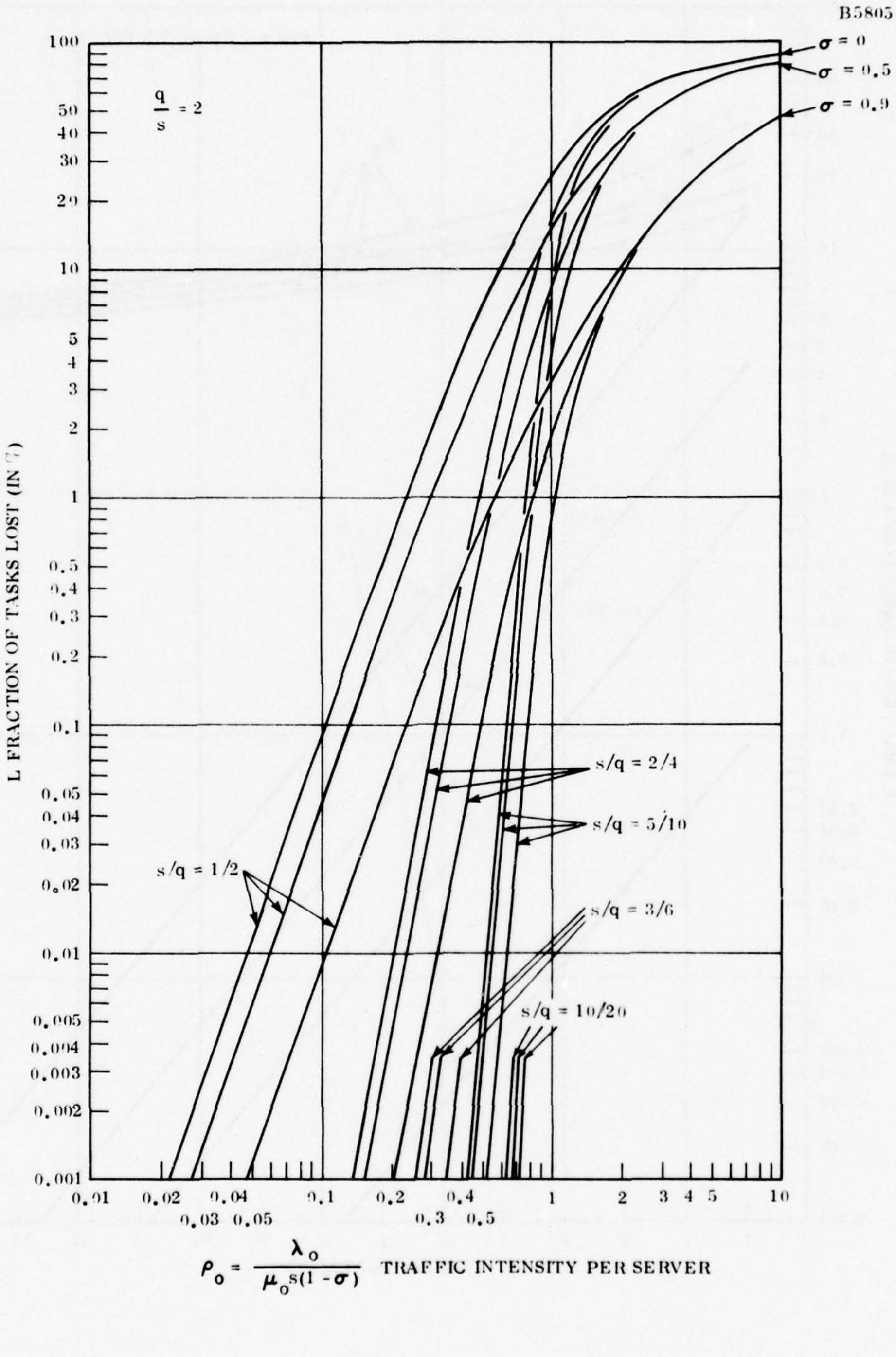

Figure 6-2. Fraction of Lost Tasks; q/s = 2

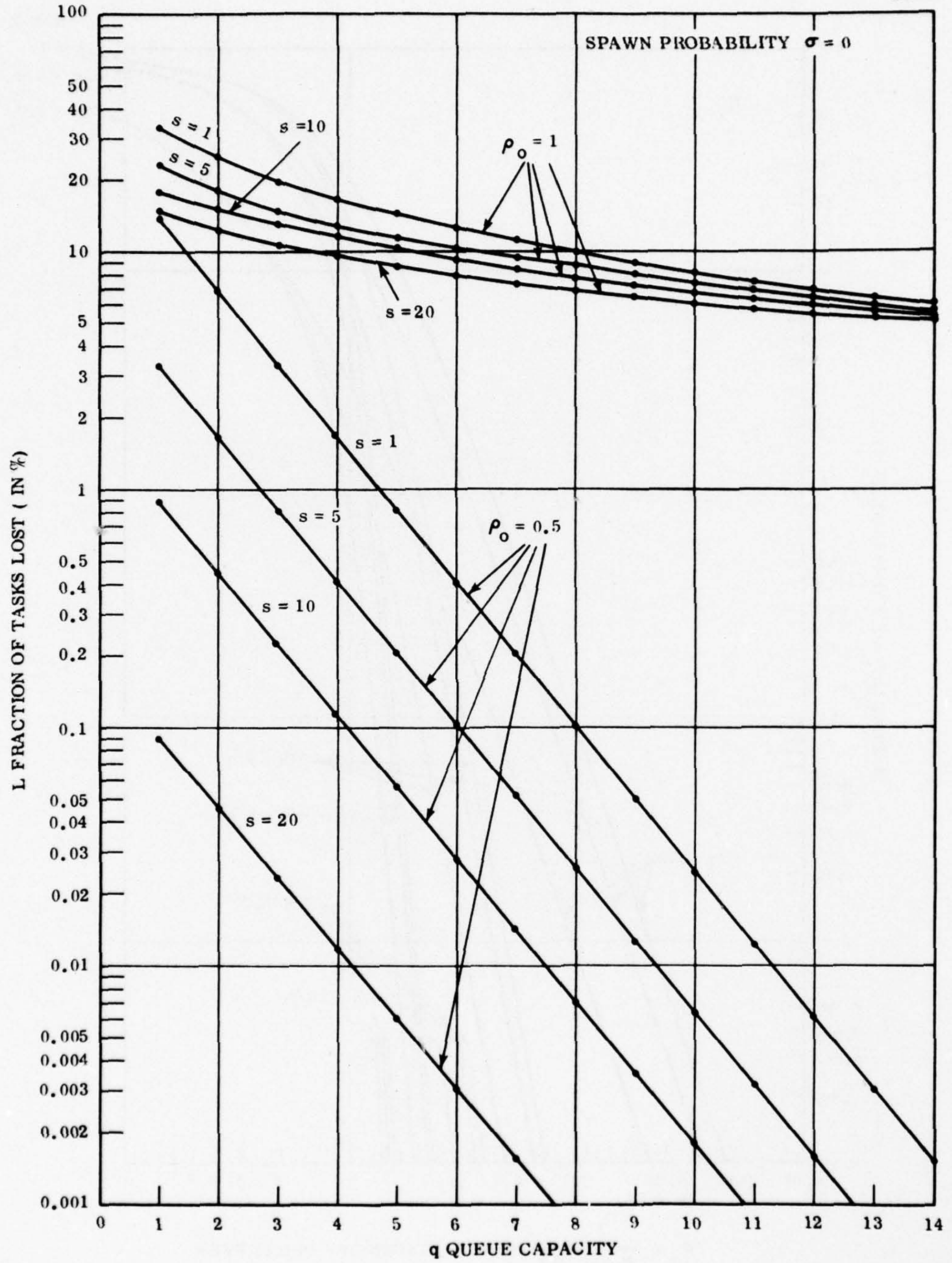

Figure 6-3. Fraction of Lost Tasks Vs Queue Capacity; $\sigma = 0$

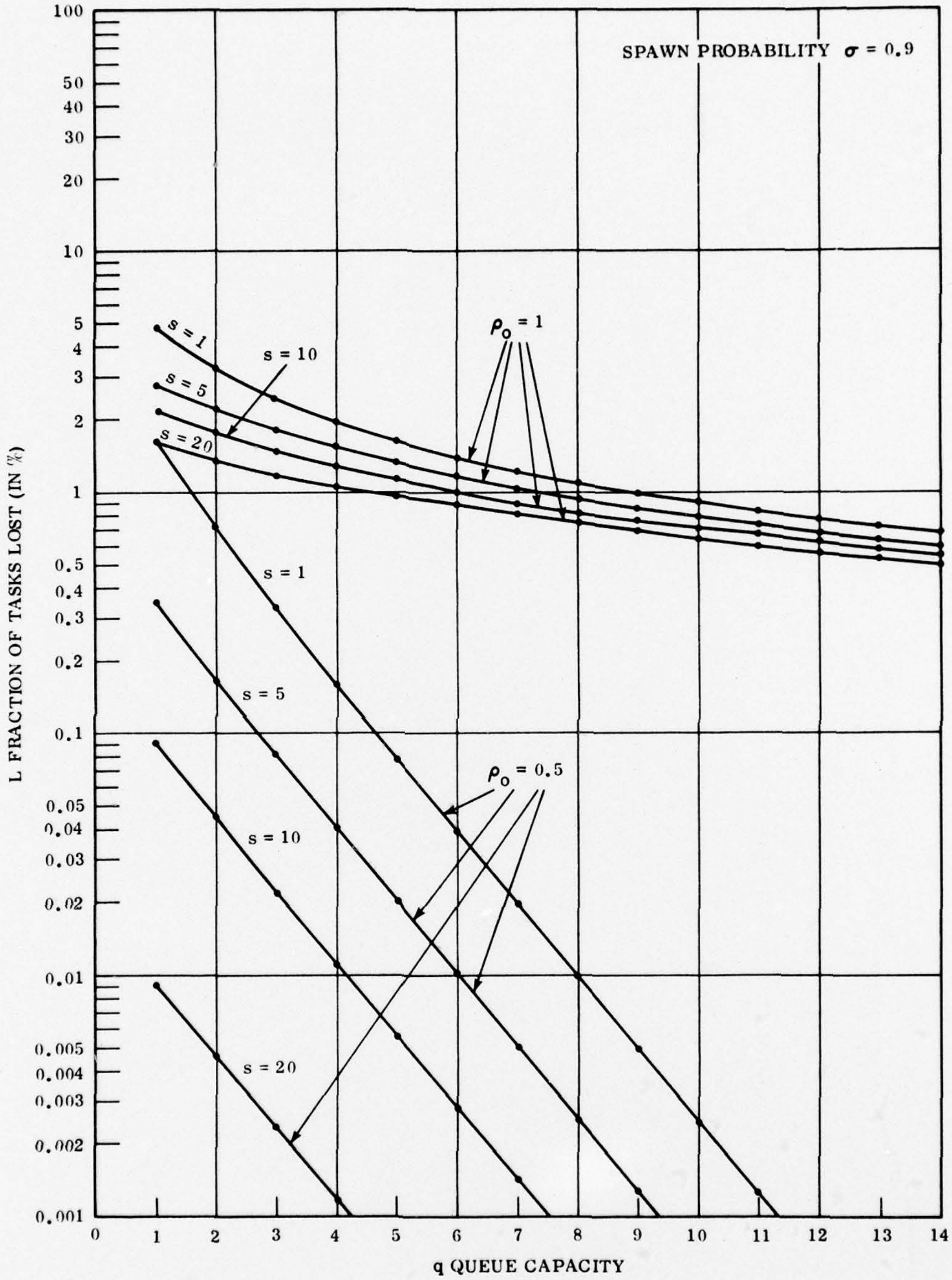

Figure 6-4. Fraction of Lost Tasks Vs Queue Capacity; σ = 0.9

Preceding Page Blank - Not Filmed

SECTION VII

BUSY PERIOD

The system is called idle if no task is in service, that is, if all servers are idle — otherwise the system is called busy. The system passes through alternating cycles of busy and idle periods. A busy period commences when a task arrives at an idle system and ends when a completed task leaves behind an idle system.

Let B be the mean duration of a busy period and I be the mean duration of an idle period. The probability that the system is idle is p_o, given by Equation (3-9). This can also be interpreted as the fraction of time that the system is idle, that is

$$p_o = \frac{I}{B + I} \tag{7-1}$$

Clearly

$$1 - p_o = \frac{B}{B+I} = \sum_{k=1}^{n} p_k \tag{7-2}$$

is the probability that the system is busy (i.e., not in the idle state) and thus the fraction of time that the system is busy.

Simply because the system is busy does not imply that all the servers are busy. In fact it just takes one server busy in order for the system to be busy. A measure of the server utilization is the <u>fraction of busy servers</u>, $\overline{N}_s/s$. Figure 7-1 shows a plot of the fraction of time the system is busy versus the fraction of busy servers for different numbers of servers. As can be seen from the graphs, for lightly loaded systems we have

$$\frac{B}{B+I} \approx s\left(\frac{\overline{N}_s}{s}\right) = N_s \qquad (\text{small } \overline{N}_s/s) \tag{7-3}$$

The mean idle period is the mean time that the system is in state 0. This is merely

$$I = \overline{\tau}_o = \frac{1}{\lambda_o} \tag{7-4}$$

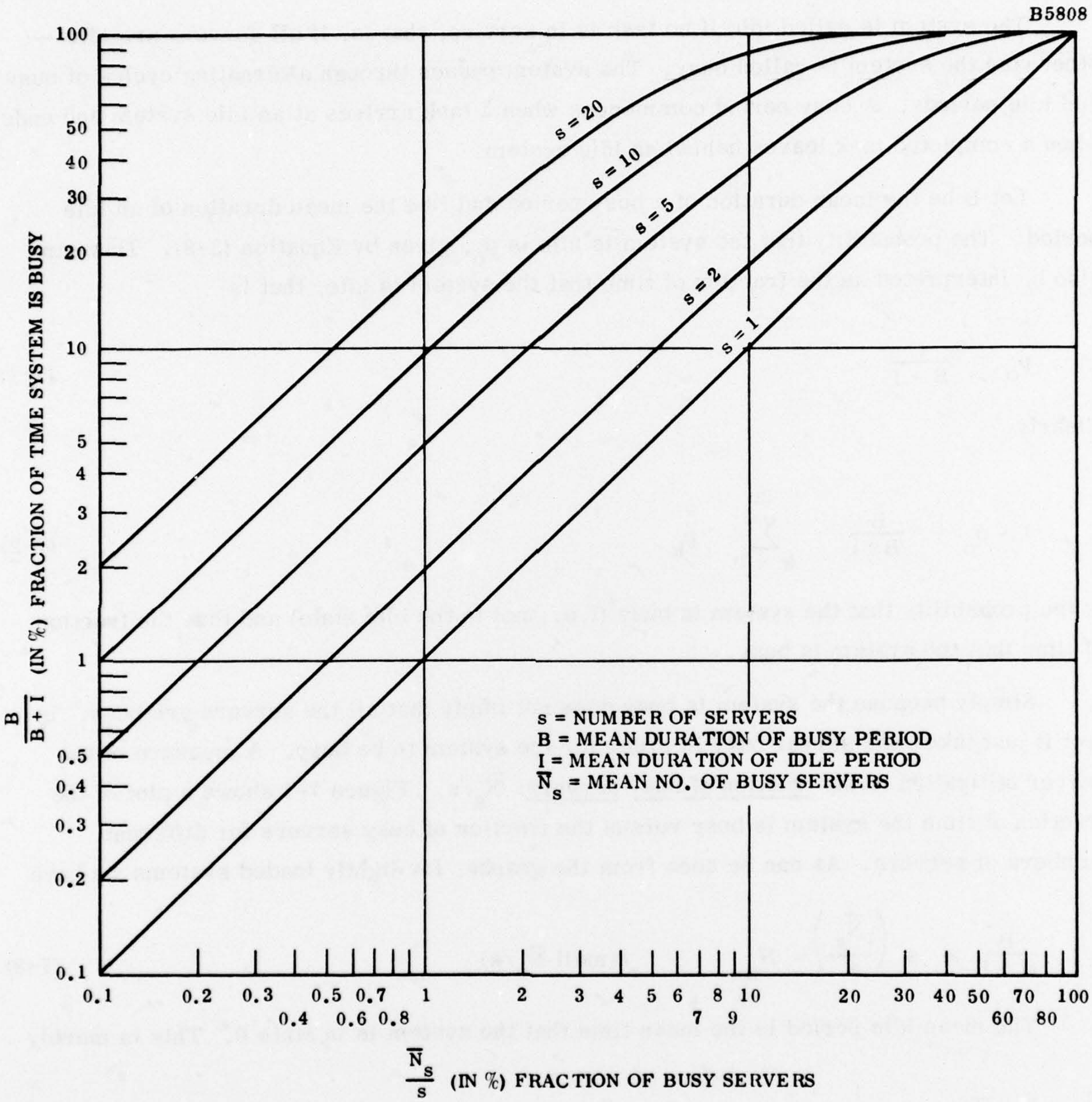

Figure 7-1. Fraction of Time System is Busy Vs Fraction Busy Servers

This may then be substituted into Equation (7-1) and solved for the busy period B:

$$B = \frac{1}{\lambda_o}\left(\frac{1 - p_o}{p_o}\right) \tag{7-5}$$

and normalizing the busy period with respect to the service time yields:

$$\mu_o B = \frac{1}{r_o(1-\sigma)}\left(\frac{1-p_o}{p_o}\right) \tag{7-6}$$

p_o is given by Equation (3-9).

Figures 7-2, 7-3, and 7-4 show plots of the normalized busy period $\mu_o B$ versus the traffic intensity per server, ρ_o. Figure 7-2 shows a family of curves for various values of queue capacity q and number of servers s for queue capacity to server ratio $q/s = 0.1$. Similarly, Figure 7-3 is for the case $q/s = 1$ and Figure 7-4 for $q/s = 10$.

The busy period will now be derived by an alternate method which is included here for interest. It uses the notion of an imbedded process. The random process N(t), which is the number of tasks in the system at time t, is defined on a continuous time scale and is, in fact, a Markov process. A Markov process has the property that for any $t > t_1 > t_2 > \dots t_m$ and $m \geq 1$

$$p\{N(t) \mid N(t_1), N(t_2) \dots N(t_m)\} = p\{N(t) \mid N(t_1)\} \tag{7-7}$$

As time progresses the system moves from one state to another. Upon arrival of a task the system moves from state k to k + 1; upon departure it moves from k to k - 1. Each time the process enters a state, it remains in that state for a period of time (which is random) and then moves to a neighboring state. It is not difficult to show that the time the process resides in a state before moving out of it is exponentially distributed. The mean time is state k, denoted by $\bar{\tau}_k$, is the reciprocal of the exit rates from that state. For example if $0 < k \leq s$ then all k tasks are in service and none in queue. The departure from state k occurs either by a new arrival to the system (rate λ_o) or by completion of a non-spawning task service (rate $k\mu_o (1 - \sigma)$).

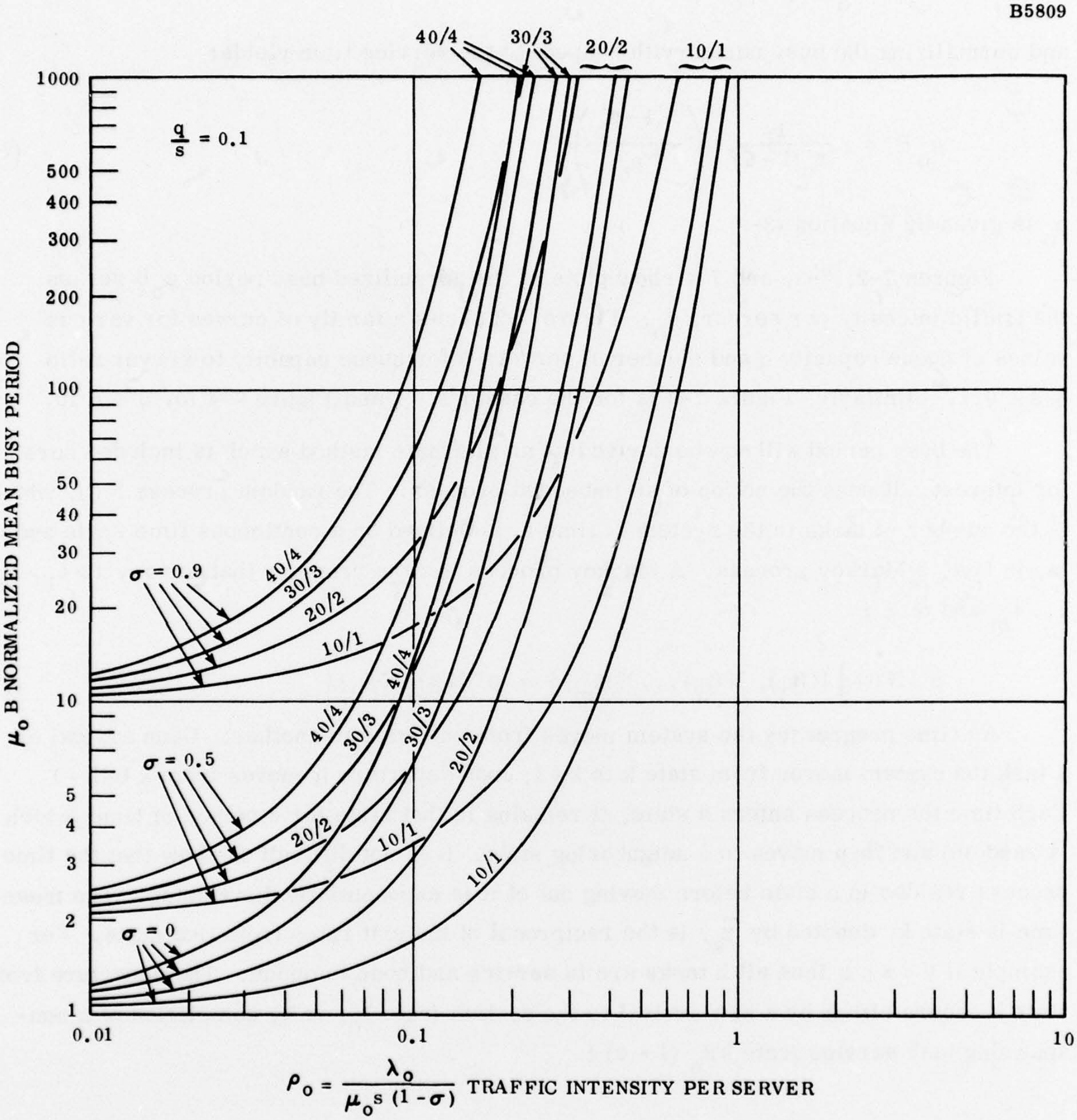

Figure 7-2. Mean Busy Period; q/s = 0.1

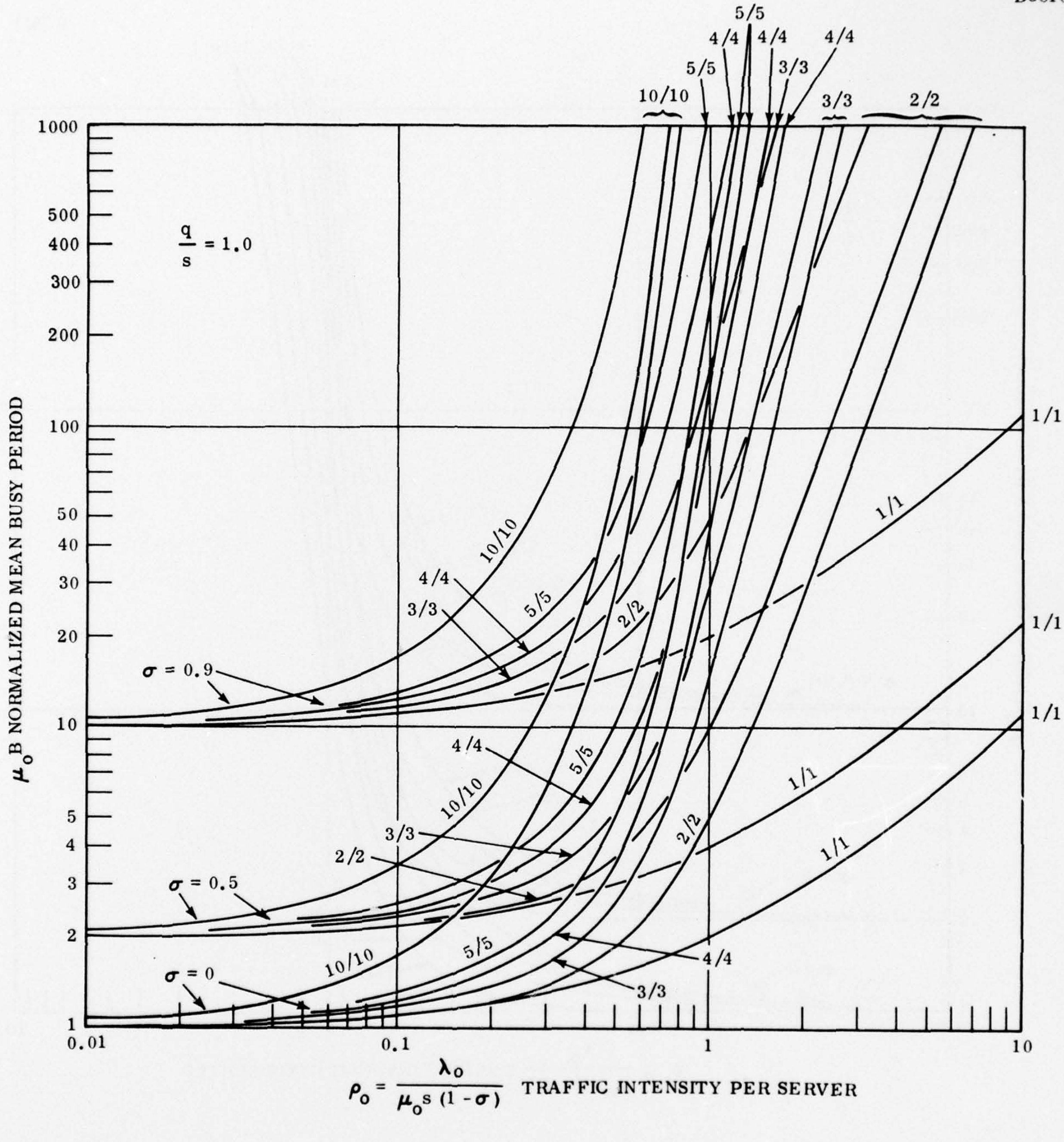

Figure 7-3. Mean Busy Period; q/s = 1.0

B5811

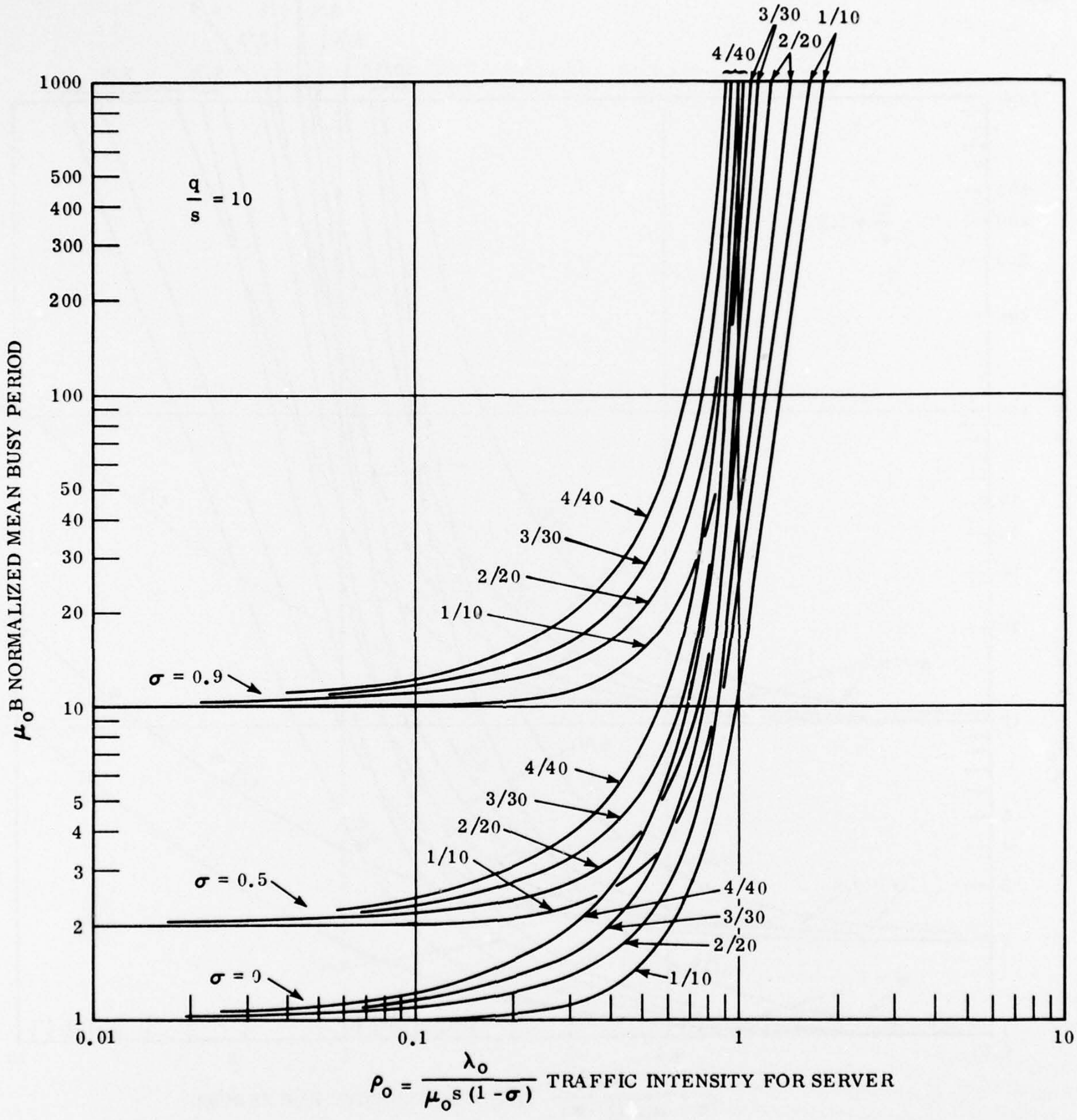

Figure 7-4. Mean Busy Period; q/s = 10

Thus, the mean time in state k is

$$\bar{\tau}_k = \begin{cases} \dfrac{1}{\lambda_o} = \dfrac{1}{\mu_o} \dfrac{1}{r_o(1-\sigma)} & k = 0 \\[2ex] \dfrac{1}{\lambda_o + k\mu_o(1-\sigma)} = \dfrac{1}{\mu_o} \dfrac{1}{(r_o+k)(1-\sigma)} & 0 < k \leq s \\[2ex] \dfrac{1}{\lambda_o + s\mu_o(1-\sigma)} = \dfrac{1}{\mu_o} \dfrac{1}{(r_o+s)(1-\sigma)} & s < k < n \\[2ex] \dfrac{1}{s\mu_o(1-\sigma)} = \dfrac{1}{\mu_o} \dfrac{1}{s(1-\sigma)} & k = n \end{cases} \tag{7-8}$$

Now we consider the system at its transition points when it changes state. Let N_i be the state the system enters at the i^{th} transition. If the i^{th} transition occurs at time t_i then clearly

$$N_i = N(t_i)$$

Let π_{km} be the transition probability that the system will enter state k given it was in state m at the previous transition:

$$\pi_{km} = P\{N_i = k \mid N_{i-1} = m\} \tag{7-9}$$

Clearly π_{km} is the exit flow rate from m to k divided by the total flow rate out of state m, i.e.:

$$\pi_{k+1,k} = \begin{cases} 1 & k = 0 \\[2ex] \dfrac{\lambda_o}{\lambda_o + k\mu_o(1-\sigma)} = \dfrac{r_o}{r_o+k} & 0 < k \leq s \\[2ex] \dfrac{\lambda_o}{\lambda_o + s\mu_o(1-\sigma)} = \dfrac{r_o}{r_o+s} & s < k < n \\[2ex] 0 & k = n \end{cases} \tag{7-10}$$

$$
\pi_{k+1,k} = \begin{cases} 0 & k = 0 \\ \dfrac{k\mu_o(1-\sigma)}{\lambda_o + k\mu_o(1-\sigma)} = \dfrac{k}{r_o + k} & 0 < k \leq s \\ \dfrac{s\mu_o(1-\sigma)}{\lambda_o + s\mu_o(1-\sigma)} = \dfrac{s}{r_o + s} & s < k < n \\ 1 & k = n \end{cases} \tag{7-11}
$$

$$
\pi_{km} = 0 \text{ if } m > k+1 \text{ or } m < k-1 \tag{7-12}
$$

Let T_{km} be the mean time it takes the system to first enter state k measured from the time the system first entered state m. It follows from this definition that the idle and busy periods are given by:

$$
\begin{aligned} I &= T_{10} \\ B &= T_{01} \end{aligned} \tag{7-13}
$$

T_{km} is the solution to the difference equations

$$
T_{km} = \bar{\tau}_m + \sum_{k \neq j} T_{kj}\pi_{jm} \tag{7-14}
$$

Equation (7-14) is not difficult to derive. Essentially, it says that the mean time from first entrance to state m until first entrance to state k is the mean time spent in state m plus the mean time to get to state k from the state the system goes to (state j) after state m, averaged over j. Using Equations (7-10) and (7-11), Equation (7-10) becomes:

$$
\begin{aligned} T_{km} &= \bar{\tau}_m + T_{k,m+1}\pi_{m+1,m} + T_{k,m+1}\pi_{m-1,m} && k \neq m+1,\ k \neq m-1 \\ &= \bar{\tau}_m + T_{m+1,m-1}\pi_{m-1,m} && k = m+1 \\ &= \bar{\tau}_m + T_{m-1,m+1}\pi_{m+1,m} && k = m-1 \end{aligned} \tag{7-15}
$$

In particular, setting k = 0, we find

$$T_{0m} = \bar{\tau}_m + T_{0,m+1}\pi_{m+1,m} + T_{0,m-1}\pi_{m-1,m} \qquad m > 1 \tag{7-16}$$

$$= \bar{\tau}_1 + T_{02}\,\pi_{21} \qquad m = 1$$

$$= \bar{\tau}_0 + T_{01}\,\pi_{10} \qquad m = 0$$

To solve these equations for the busy period $B = T_{01}$ we set m = n and obtain (using Equations (7-10) and (7-11))

$$T_{on} = \bar{\tau}_n + T_{o,n-1} \tag{7-17}$$

This equation may be substituted into the next lower index equation and solved for $T_{o,n-1}$. The procedure, performed recursively, yields at each stage an equation of the form

$$\begin{aligned} T_{ok} &= \alpha_k + \beta_k\, T_{o,k-1} && k \neq 1 \\ &= \alpha_1 && k = 1 \end{aligned} \tag{7-18}$$

where

$$\alpha_k = \frac{\bar{\tau}_k + \pi_{k+1,k}\alpha_{k+1}}{1 - \pi_{k+1,k}\beta_{k+1}} \tag{7-19}$$

$$\beta_k = \frac{\pi_{k-1,k}}{1 - \pi_{k+1,k}\beta_{k+1}}$$

and starting with

$$\begin{aligned} \alpha_n &= \bar{\tau}_n \\ \beta_n &= 1 \end{aligned} \tag{7-20}$$

Thus the solution of Equation (7-16) for T_{01} is obtained by evaluating Equation (7-19) recursively, starting with Equation (7-20), obtaining finally the mean busy period $B = T_{01} = \alpha_1$.

Preceding Page Blank - Filmed

SECTION VIII

TASK PARTITIONING AND SPAWNING

The application for which this analysis was performed involved a multi-microprocessor system with queue memories. The servers represent processors and the tasks represent requests for processor time. The spawning feature of the model is intended to reflect the partitioning of a job into smaller tasks. The definition of task size is the responsibility of the system designer. Should the system be designed with large tasks so that each job, once it enters the server, is processed to completion and then leaves the system; or should tasks be defined small so that upon each task completion the job may have to rejoin the queue and thus continue circulating (i.e., spawning) through the system until it is completed?

Some of the tradeoffs are clear. Each time a task circulates through the system it must wait in queue before being processed. Therefore, if the overhead per service or the response time for a job is critical it is not desirable to design a system with small tasks and hence, high spawning. On the other hand, with high traffic loads, lost tasks can become critical due to the queue being full upon arrival of a new task. In this regime, high spawning will decrease the lost task rate since spawning is synchronized with a decrease in queue length as discussed earlier.

As a matter of nomenclature we shall refer to a newly arriving task and all of the spawned tasks that have originated from this present task as a job. Therefore the job arrival rate is the external task arrival rate, λ_o. Suppose a task spawns k additional tasks in sequence (that is, the parent task spawns a task which circulates back through the system; this in turn spawns another task, and so on for k times). Then we can write

$$\tau = \tau_0 + \tau_1 + \tau_2 + \ldots + \tau_k \tag{8-1}$$

where τ_0 is the service time for the parent task and τ_i, $i = 1, 2, \ldots k$, is the service time for the i^{th} spawned task. τ is the total service time for the job.*

k is a discrete random variable denoting the number of spawns a parent task generates. Let p(k) be the probability that a parent spawns k tasks. Then $p(0) = 1 - \sigma$ and

$$p(k) = (1 - \sigma)\sigma^k \qquad k = 0, 1, 2, 3 \ldots \tag{8-2}$$

---

* In this section τ_i denotes the service time for a spawned task and is a random variable. in the previous section we used the notation $\bar{\tau}_i$ to denote the mean time the system is in state i. These quantities are quite different and should not be confused.

The mean number of spawns is

$$\bar{k} = \sum_{k=0}^{\infty} k\,p(k) \qquad (8\text{-}3)$$

$$= \sigma/(1-\sigma)$$

The job service time τ, given by Equation (8-1), is the sum of a random number of random variables. From the assumptions of the model each τ_i is exponentially distributed with mean $1/\mu_o$:

$$f_i(\tau_i) = \mu_o\, e^{-\mu_o \tau_i} \qquad i = 0, 1, 2, \ldots k \qquad (8\text{-}4)$$

$$\tau_i \geq 0$$

Therefore the characteristic function of τ_i is

$$g_i(z_i) = \frac{\mu_o}{z_i - \mu_o} \qquad i = 0, 1, 2, \ldots k \qquad (8\text{-}5)$$

To find $f(\tau)$, the distribution of τ, we have, from Bayes' theorem,

$$f(\tau) = \sum_{k=0}^{\infty} f(\tau \mid k)\, p(k) \qquad (8\text{-}6)$$

and the characteristic function of $f(\tau)$ is given by:

$$g(z) = \sum_{k=0}^{\infty} g(z \mid k)\, p(k) \qquad (8\text{-}7)$$

But $f(\tau \mid k)$ is the distribution of τ given k, thus it is the k-fold convolution of $f_i(\tau)$. Consequently $g(z \mid k)$ is merely $g_i(z)$ raised to the k^{th} power:

$$g(z \mid k) = \left(\frac{\mu_o}{z - \mu_o}\right)^k \qquad (8\text{-}8)$$

Putting Equations (8-2) and (8-8) into (8-7) yields

$$g(z) = \frac{\mu_o(1-\sigma)}{z - \mu_o(1-\sigma)} \tag{8-9}$$

from which we conclude that τ is also exponentially distributed:

$$f(z) = \mu_o(1-\sigma)\, e^{-\mu_o(1-\sigma)\tau} \tag{8-10}$$

Thus the mean service time for a job is

$$\bar{\tau} = \frac{1}{\mu_o(1-\sigma)} \tag{8-11}$$

External jobs or tasks arrive at the system at a rate λ_o and the average work that arrives per unit time is

$$\lambda_o \bar{\tau} = \frac{\lambda_o}{\mu_o(1-\sigma)} = r_o \tag{8-12}$$

and the traffic intensity per server is

$$\rho_o = \frac{\lambda_o}{s} \bar{\tau} \tag{8-13}$$

Suppose we wish to study the effect on the system due to varying the spawn probability σ, but keeping the mean job service time $\bar{\tau}$, the mean job arrival rate λ_o and the number of servers s constant. In this case ρ_o and r_o remain constant. For example, Figures 6-1 and 6-2 show plots of the fraction of lost tasks L versus ρ_o for different values of σ, s and q. Letting $r_o = \lambda_o \bar{\tau} = 0.2$, $q = s = 2$ and $\rho_o = 0.1$, then we have from Figure 6-1:

$$L = \begin{cases} 0.0164\% & \text{at } \sigma = 0,\ \bar{k} = 0 \\ 0.0082\% & \text{at } \sigma = 0.5,\ \bar{k} = 1 \\ 0.0016\% & \text{at } \sigma = 0.9,\ \bar{k} = 9 \end{cases}$$

This shows the effect on lost tasks by partitioning jobs into smaller tasks. The case $\sigma = 0$ implies, from Equation (8-3), that $\bar{k} = 0$, that is, there is no spawning and thus the mean task size is equal to the mean job size. The case $\sigma = 0.5$, $\bar{k} = 1$ means that on the average the job requires one spawn, i.e., the average task size is half the average job size. Similarly for $\sigma = 0.9$, $\bar{k} = 9$, the average task size is one tenth the average job size, thus requiring nine spawns.

Preceding Page Blank - NOT FILMED

REFERENCES

1. D.R. Mott and G. Arabadjis, "Multimicroprocessor with Queue Memories", General Electric Co., Syracuse, N.Y., July 1977, ADE Tech Memo 195.

2. L. Klienrock, Queueing Systems Volume 1: Theory, John Wiley & Sons, Inc., 1975, p. 17.